ATARI® 400/800™

ATARI® HOME COMPUTER SYSTEM

TECHNICAL REFERENCE NOTES

includes:

Operating System User's Manual
Operating System Source Listing
and
Hardware Manual

TO ALL PERSONS RECEIVING THIS DOCUMENT

Reproduction is forbidden without the specific written permission of ATARI, INC. Sunnyvale, CA 94086. No right to reproduce this document, nor the subject matter thereof, is granted unless by written agreement with, or written permission from the Corporation.

ATARI®
A Warner Communications Company

C016555 Rev. A

ATARI® 400/800™

ATARI® HOME COMPUTER SYSTEM

OPERATING SYSTEM USER'S MANUAL

ATARI®
A Warner Communications Company

COPYRIGHT 1982, ATARI, INC.
ALL RIGHTS RESERVED

TO ALL PERSONS RECEIVING THIS DOCUMENT

Reproduction is forbidden without the specific written permission of ATARI, INC. Sunnyvale, CA 94086. No right to reproduce this document, nor the subject matter thereof, is granted unless by written agreement with, or written permission from the Corporation.

Every effort has been made to ensure that this manual accurately documents this product of the ATARI Home Computer Division. However, due to the ongoing improvement and update of the computer software and hardware, ATARI, INC. cannot guarantee the accuracy of printed material after the date of publication and disclaims liability for changes, errors, or omissions.

PREFACE 17

1 INTRODUCTION 18

GENERAL DESCRIPTION
OF THE ATARI COMPUTER SYSTEM 18

Conventions Used in This Manual 20

HEXADECIMAL NUMBERS 20
MEMORY ADDRESSES 20
KILOBYTES OF MEMORY 20
PASCAL AS AN ALGORITHM-SPECIFICATION LANGUAGE 20
MEMORY LAYOUTS 20
BACKUS-NAUR FORM (BNF) 21
OS-EQUATE FILENAMES 21

2 OPERATING SYSTEM FUNCTIONAL ORGANIZATION 22

Input/Output Subsystem 22
Interrupt Processing 22
Initialization 22

Power-Up 22
System Reset 23

Floating Point Arithmetic Package 24

3 CONFIGURATIONS 25

Program Environments 25

Blackboard Mode 25
Cartridge 26
Diskette-Boot 26
Cassette-Boot 26

RAM Expansion 27
Peripheral Devices 27

Game Controllers 27
Program Recorder 27
Serial Bus Devices 28

4 SYSTEM MEMORY UTILIZATION 29

RAM Region 29

Page 0 30
Page 1 30
OS Data Base 30
User Workspace 31
Boot Region 31
Screen Display List and Data 31
Free Memory Region 31

Cartridges A and B 31
Mapped I/O 32
Resident OS and Floating Point Package ROM 32
Central Data Base Description 32
Memory Dynamics 32

System Initialization Process 33
Changing Screen Modes 33

5 I/O SUBSYSTEM 34

Central I/O Utility 36
CIO Design Philosophy 37

DEVICE INDEPENDENCE 37
DATA ACCESS METHODS 37
MULTIPLE DEVICE/FILE CONCURRENCY 38
UNIFIED ERROR HANDLING 38
DEVICE EXPANSION 38

CIO CALLING MECHANISM 38

HANDLER ID -- ICHID [0340] 39
DEVICE NUMBER -- ICDNO [0341] 39
COMMAND BYTE -- ICCMD [0342] 40
STATUS -- ICSTA [0343] 40
BUFFER ADDRESS
ICBAL[0344] AND ICBAH [0345] 40
PUT ADDRESS --
ICPTL [0346] AND ICPTH [0347] 40
BUFFER LENGTH/BYTE COUNT --
ICBLL [034B] and ICBLH [0349] 40
AUXILIARY INFORMATION --
ICAX1 [034A] and ICAX2 [034B] 40
REMAINING BYTES (ICAX3-ICAX6) 41

CIO Functions 41

OPEN -- Assign Device/Filename to IOCB and Ready for Access 41
CLOSE -- Terminate Access to Device/File and Release IOCB 42
GET CHARACTERS -- Read n Characters (Byte-Aligned Access) 43
PUT CHARACTERS -- Write n Characters (Byte-Aligned Access) 43
GET RECORD -- Read Up To n Characters (Record-Aligned Access) 44
PUT RECORD -- Write Up To n Characters (Record-Aligned Access) 44
GET STATUS -- Return Device-Dependent Status Bytes 45
SPECIAL -- Special Function 45

Device/Filename Specification 46
I/O Example 47

Device Specific Information 50
Keyboard Handler 50

CIO Function Descriptions 51

Theory of Operation 51

Display Handler (S:) 54

Screen Modes 54
TEXT MODE 0 54
TEXT MODES 1 AND 2 55
GRAPHICS MODES (Modes 3 Through 11) 56
SPLIT-SCREEN CONFIGURATIONS 56

CIO Function Descriptions 57
User-Alterable Data Base Variables 61
Theory of Operation 62

Screen Editor (E:) 66

CIO Function Descriptions 67
User-Alterable Data Base Variables 70

Cassette Handler (C:) 72
CIO Function Descriptions 72

Theory of Operation 74
File Structure 75

Printer Handler (P:) 76
CIO Function Descriptions 76
Theory of Operation 78

Disk File Manager (D:) 78

CIO Function Descriptions 79
Device/Filename Specification 81

Filename Wildcarding 82

Special CIO functions 84
Theory of Operation 87
FMS Diskette Utilization 89

FMS BOOT RECORD FORMAT 90
BOOT PROCESS MEMORY MAP 92
VOLUME TABLE OF CONTENTS 93
FILE DIRECTORY FORMAT 94
FMS FILE SECTOR FORMAT 95

Non-CIO I/O 96
Resident Device Handler Vectors 96

Resident Diskette Handler 97
Diskette Handler Commands 99

Serial Bus I/O 101

6 INTERRUPT PROCESSING 102

Chip-Reset 103
Nonmaskable Interrupts 103

Stage 1 VBLANK Process 104
Stage 2 VBLANK Process 105

Maskable Interrupts 107
Interrupt Initialization 108
System Timers 109
Usage Notes 109

POKEY Interrupt Mask 110
Setting Interrupt and Timer Vectors 110
Stack Content at Interrupt Vector Points 111
Miscellaneous Considerations 112

Flowcharts 113

7 SYSTEM INITIALIZATION 116

Power-Up Initialization (Coldstart) Procedure 116
System Reset Initialization (Warmstart) Procedure 119

8 FLOATING POINT ARITHMETIC PACKAGE 121

Functions/Calling Sequences 122

ASCII to Floating Point Conversion (AFP) 122
Floating Point to ASCII Conversion (FASC) 122
Integer to Floating Point Conversion (IFP) 123
Floating Point to Integer Conversion (FPI) 123
Floating Point Addition (FADD) 124
Floating Point Subtraction (FSUB) 124
Floating Point Multiplication (FMUL) 124
Floating Point Division (FDIV) 125
Floating Point Logarithms (LOG and LOG10) 125
Floating Point Exponentiation (EXP and EXP10) 126
Floating Point Polynomial Evaluation (PLYEVL) 126
Clear FR0 (ZFR0) 127
Clear Page-Zero Floating Point Number (ZF1) 127
Load Floating Point Number to FR0
(FLD0R and FLD0P) 127
Load Floating Point Number to FR1
(FLD1R and FLD1P) 128
Store Floating Point Number From FR0
(FSTOR and FSTOP) 128
Move Floating Point Number From FR0 to FR1
(FMOVE) 128

Resource Utilization 128
Implementation Details 129

9 ADDING NEW DEVICE HANDLERS/PERIPHERALS 131

Device Table 134
CIO/Handler Interface 134

Calling Mechanism 135
Handler Initialization 136
Functions Supported 136
Error Handling 140
Resource Allocation 140

ZERO-PAGE RAM 141
NONZERO-PAGE RAM 141
STACK SPACE 142

Handler/SIO Interface 142

Calling Mechanism 142
Functions Supported 144
Error Handling 144

Serial I/O Bus Characteristics and Protocol 145

Hardware/Electrical Characteristics 145
Serial Port Electrical Specifications 147
Bus Commands 147

COMMAND FRAME 148
COMMAND FRAME ACKNOWLEDGE 148
DATA FRAME 149
OPERATION COMPLETE 149

Bus Timing 150

Handler Environment 152

Bootable Handler 153
Cartridge Resident Handler 153
Flowcharts 153

10 PROGRAM ENVIRONMENT AND INITIALIZATION 157

Cartridge 157

Cartridge Without Booted Support Package 158
Cartridge With Booted Support Package 158

Diskette-Booted Software 159

Diskette-Boot File Format 159
Diskette-Boot Process 160
Sample Diskette-Bootable Program Listing 161
Program to Create Diskette-Boot Files 162

Cassette-Booted Software 164

Cassette-Boot File Format 165
Cassette-Boot Process 165
Sample Cassette-Bootable Program Listing 167
Program to Create Cassette-Boot Files 168

11 ADVANCED TECHNIQUES AND APPLICATION NOTES 170

Sound Generation 170

Capabilities 170
Conflicts With OS 170

Screen Graphics 171

Hardware Capabilities 171
OS Capabilities 171
Cursor Control 171
Color Control 171
Alternate Character Sets 172

Player/Missile Graphics 174

Hardware Capabilities 174
Conflicts With OS 174

Reading Game Controllers 174

Keyboard Controller Sensing 174
Front Panel Connectors as I/O Ports 176

Hardware Information: 176
Software Information: 177
Other Miscellaneous Software Information: 179

APPENDICES

Appendix A -- CIO COMMAND BYTE VALUES 180

Appendix B -- CIO STATUS BYTE VALUES. 181

Appendix C -- SIO STATUS BYTE VALUES 182

Appendix D -- ATASCII CODES 183

Appendix E -- DISPLAY CODES (ATASCII) 184

Appendix F -- KEYBOARD CODES (ATASCII) 185

Appendix G -- PRINTER CODES (ATASCII) 186

Appendix H -- SCREEN MODE CHARACTERISTICS 188

Appendix I -- SERIAL BUS ID AND COMMAND SUMMARY 191

Appendix J -- ROM VECTORS 192

Appendix K -- DEVICE CHARACTERISTICS 194

Keyboard 194
Display 194
ATARI 410[TM] Program Recorder 194
ATARI 820[TM] 40-Column Impact Printer 195
ATARI 810[TM] Disk Drive 197

Appendix L -- OS DATA BASE VARIABLE FUNCTIONAL DESCRIPTIONS 200

Central Data Base Description 200

FUNCTIONAL INDEX TO DATA BASE VARIABLE DESCRIPTIONS 201

A. MEMORY CONFIGURATION 211

B. TEXT/GRAPHICS SCREEN 212

Cursor Control 212
Screen Margins 213
Text Scrolling 215
Attract Mode 215
Tabbing 216
Logical Text Lines 217
Split Screen 218
Displaying Control Characters 220
Escape (Display Following Control Character) 221
Display Control Characters Mode 221
Bit-Mapped Graphics 221
Internal Working Variables 222
Internal Character Code Conversion 224

C. DISKETTE HANDLER 225

D. CASSETTE 225

Baud Rate Determination 226
Cassette Mode 227
Cassette Buffer 227
Internal Working Variables 228

E. KEYBOARD 229

Key Reading and Debouncing 229
Special Functions 230
Start/Stop 230
Autorepeat 231
Inverse Video Control 232
Console Keys: [SELECT], [START], and [OPTION] 232

F. PRINTER 232

Printer-Buffer 233
Internal Working Variables 233

G. CENTRAL I/O ROUTINE (CIO) 233

User Call Parameters 233
I/O Control Block 233
Device Status 234
Device Table 235
CIO/Handler Interface Parameters 235
Zero-Page IOCB 235
Internal Working Variables 236

H. SERIAL I/O ROUTINE (SIO) 237

User Call Parameters 237
Device Control Block 237
Bus Sound Control 238
Serial Bus Control 238
Retry Logic 238
Checksum 239
Data Buffering 240
General Buffer Control 240
Command Frame Output Buffer 240
Receive/Transmit Data Buffering 241
SIO Timeout 241
Internal Working Variables 242

J. ATARI CONTROLLERS 243

Joysticks 243
Paddles 244
Light Pen 245
Driving Controllers 246

K. DISK FILE MANAGER 247

L. DISK UTILITY POINTER 248

M. FLOATING POINT PACKAGE 248

N. Power-Up and System Reset 249

RAM Sizing 249
Diskette/Cassette-Boot 250
Environment Control 251

P. INTERRUPTS 252

System Timers 253
Real Time Clock 253
System Timer 1 253
System Timer 2 254
System Timers 3, 4 and 5 254
RAM Interrupt Vectors 255
NMI Interrupt Vectors 255
IRQ Interrupt Vectors 255
Hardware Register Updates 256
Internal Working Variables 258

R. USER AREAS 258

Alphabetical List of Data Base Variables 259
Memory Address Ordered List of Data Base Variables 266
Floating Point Package Variables 270

INDEX 271

TABLE OF ILLUSTRATIONS

Figure 1-1. ATARI Home Computer Block Diagram 19
Figure 1-2. Memory Layout Chart 20

Figure 4-1. 6502 System Memory Map 29
Figure 4-2. Mapped I/O 32

Figure 5-1. I/O Subsystem Structure Flow Diagram 35
Figure 5-2. CIO Calling Mechanism 38
Figure 5-3. An I/O Example 49
Figure 5-4. Keycode to ATASCII Conversion Table 53
Figure 5-5. Text Modes 1 and 2 Data Form 56
Figure 5-6. Graphics Modes 3-11 GET Data Form 58
Figure 5-7. Graphics Modes 3-11 PUT Data Form 59
Figure 5-8. Screen Display Block Diagram 64
Figure 5-9. Cassette Handler Record Format 74
Figure 5-10. Device/Filename Syntax 81
Figure 5-11. File Management Subsystem Diskette Sector Utilization Map 89
Figure 5-12. File Management Subsystem Boot Record Format 90
Figure 5-13. File Management Subsystem Boot Process Memory Map 92
Figure 5-14. File Management Subsystem Volume Table of Contents 93
Figure 5-15. File Management Subsystem Volume Bit Map 93
Figure 5-16. File Directory Format 94
Figure 5-17. File Management Subsystem File Sector Format 95
Figure 5-18. Resident Device Handler Vectors 96
Figure 5-19. DVSTAT 4-Byte Operation Status Format 100

Figure 6-1. List of System Interrupt Events 102
Figure 6-2. Interrupt RAM Vector Initialization 108
Figure 6-3. POKEY Interrupt Mask Example 110
Figure 6-4. Interrupt and Timer Vector RAM Stack Content Table 112

Figure 9-1. I/O Subsystem Flow Diagram 133
Figure 9-2. Device Table Format 134
Figure 9-3. Handler Vector Table 135
Figure 9-4. Serial Bus Connector Pin Descriptions 146
Figure 9-5. Serial Bus Command Frame Format 148
Figure 9-6. Serial Bus Timing Diagram 151

Figure 10-1. Cartridge Header Format 157
Figure 10-2. Diskette Boot File Format 159
Figure 10-3. Diskette-Bootable Program Listing Example 162
Figure 10-4. Sample Cassette-Bootable Program 168

Figure 11-1. User-Defined Character Set Bit Memory Address 172
Figure 11-2. User-Defined 8 x 8 Character Matrix Bit Table 173
Figure 11-3. Character Base Diagram 173
Figure 11-4. Reading Data From an ATARI Keyboard Controller 176
Figure 11-5. ATARI Keyboard Controller Variable/Register Value Table 176
Figure 11-6. Using Front Panel Connectors As I/O Ports: Pin Function Tables 179

PREFACE

This manual describes the resident Operating System (OS) for the ATARI@ Home Computer, for readers who are familiar with the internal behavior of the system. It discusses:

- System functions and utilization techniques
- Subsystem relationships and organization
- Characteristics of the ATARI peripheral devices that can be attached to the ATARI400[TM] and ATARI 800[TM] Home Computer
- Advanced techniques for going beyond the basic OS capabilities
- The general features of the computer system hardware used by the OS.

It would be helpful to have a familiarity with programming concepts and terminology, assembly language programming in general, the Synertek 6502 in particular, and digital hardware concepts and terminology. you will be provided with the information you need to use the OS resources, without resorting to trial-and-error techniques or the OS listing. Supporting information for tasks that involve OS listing references is also provided.

This manual does not present a comprehensive description of the hardware used to provide OS capabilites. The programmer who needs to go beyond the capabilities described should consult the ATARI Home Computer Hardware Manual.

1 INTRODUCTION

GENERAL DESCRIPTION OF THE ATARI HOME COMPUTER SYSTEM

Operating systems in the ATARI@ 400[TM] and ATARI 800[TM] Home Computer are identical. The primary differences between the two are:

- Physical packaging
- The ATARI 400 Computer console has one cartridge slot, the ATARI 800 Computer console has two cartridge slots
- The ATARI 400 Home Computer contains 16K RAM and cannot be expanded. The ATARI 800 Home Computer can be expanded to a maximum of 48K RAM.
- The ATARI 800 Computer has a monitor jack; the ATARI 400 Computer does not.

The Hardware Circuitry

- Produces both character and point graphics for black and white (B/W) or color television.
- Produces four independent audio channels (frequency controlled) which use the television sound system.
- Provides one bi-level audio output in the base unit.
- Interfaces with up to four Joysticks and eight Paddle Controllers.
- Interfaces with a serial I/O bus for expansion.
- Contains a built-in keyboard

Figure 1-1 presents a simplified block diagram of the hardware. See the hardware manual for supporting documentation.

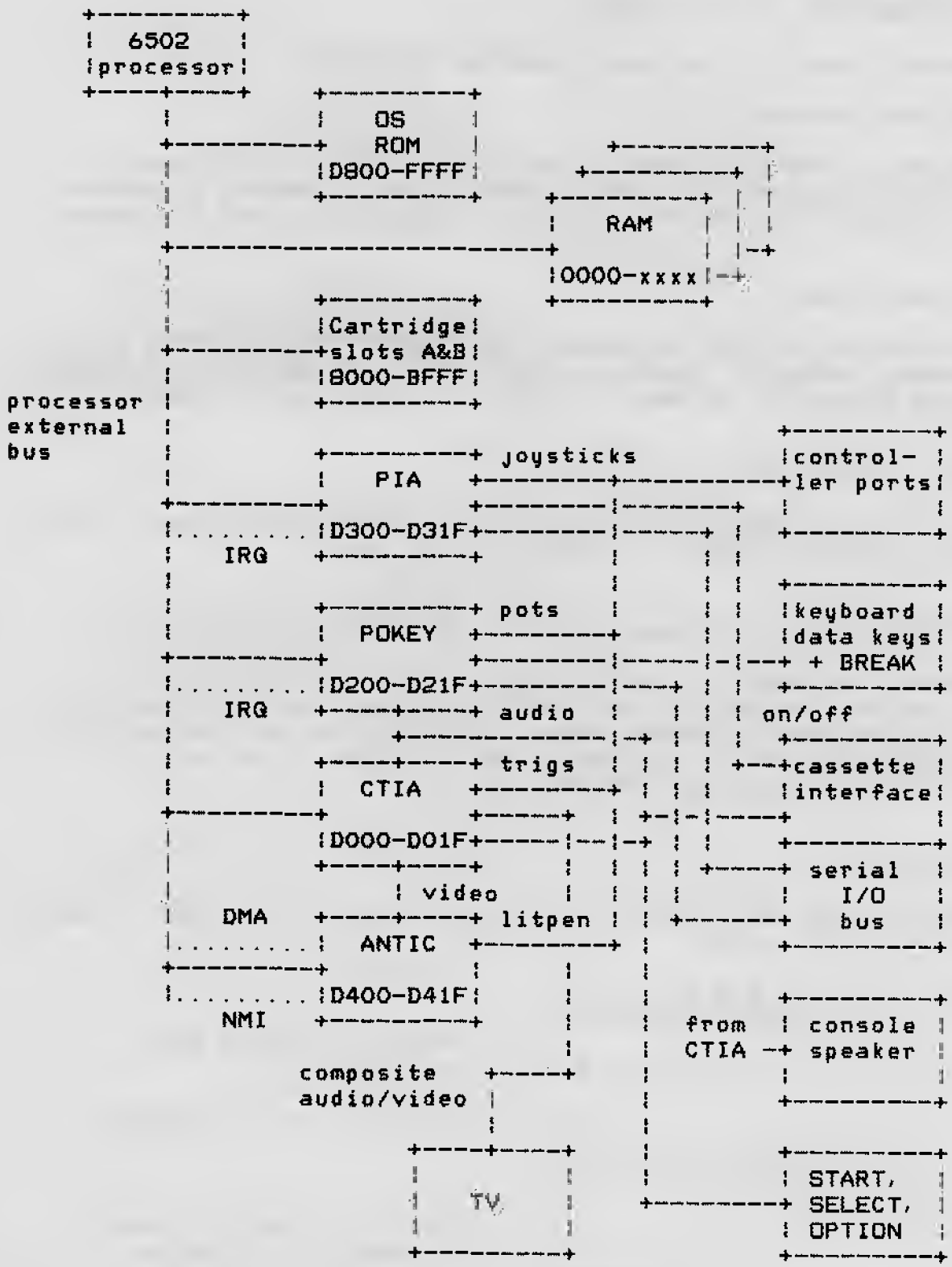

Figure 1-1. ATARI Home Computer Block Diagram

CONVENTIONS USED IN THIS MANUAL

This manual uses the following special notations:

Hexadecimal Numbers

All two-digit numbers preceded by a dollar sign ($) designate hexadecimal numbers. All other numbers (except memory addresses) are in decimal form unless otherwise specified in the supporting text.

Memory Addresses

All references to computer memory and mapped I/O locations are in hexadecimal notation. Memory addresses may or may not be contained in square brackets. (Example: [D20F] and D20F are the same address.)

Kilobytes of Memory

Memory sizes are frequently expressed in units of kilobytes, such as 32K, where a kilobyte is 1024 bytes of memory.

PASCAL As an Algorithm-Specification Language

The PASCAL language (procedure block only) is used as the specification language in the few places where an algorithm is specified in detail. PASCAL syntax is similar to any number of other block-structured languages, and you should have no difficulty following the code presented.

Memory Layouts

Diagrams similar to Figure 1-2 are used whenever pictures of bytes or tables are presented:

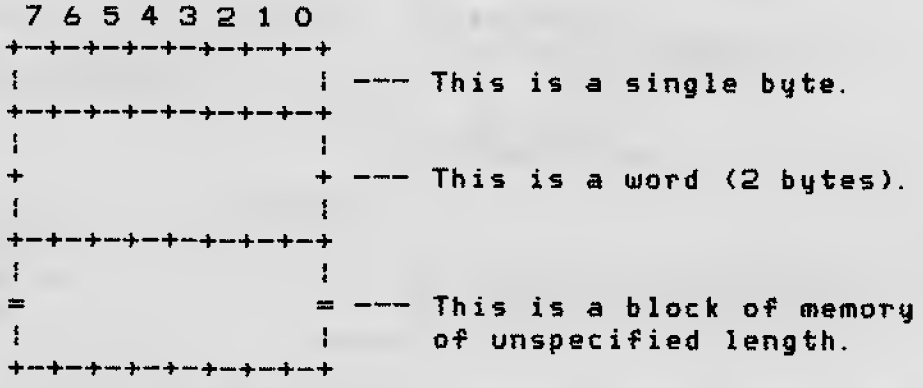

Figure 1-2. Memory Layout Chart

Bit 7 is the most significant bit (MSB) of the byte, and Bit 0 is the least significant bit (LSB).

In tables and figures, memory addresses always increase toward the bottom of the figure.

Backus-Naur Form

A modified version of Backus-Naur Form (BNF) is used to express some syntactic forms, where the following metalinguistic symbols are used:

::=	is the substitution (assignment) operator.
< >	a metasyntactic variable.
!	separates alternative substitutions.
[ ]	an optional construct.

Anything else is a syntactic literal constant, which stands for itself.

For Example:

<device specification> ::= <device name>[<device number>]:

<device name> ::= C!D!E!K!P!R!S

<device number> ::= 1!2!3!4!5!6!7!8

A "device specification" consists of a mandatory "device name," followed by an optional "device number," followed by the mandatory colon character. The device name in turn must be one of the characters shown as alternatives. The device number (if it is present) must be a digit 1 through 8.

OS Equate Filenames

Operating System ROM (Read Only Memory) and RAM (Random Access Memory) vector names, RAM database variable names and hardware register names are all referred to by the names assigned in the OS program equate list. When one of these names is used, the memory address is usually provided, such as BOOTAD [0242].

2 OPERATING SYSTEM FUNCTIONAL ORGANIZATION

This section describes the various subsystems of the resident OS in general terms.

Input/Output Subsystem

The Input/Output (I/O) subsystem provides a high-level interface between the programs and the hardware. Most functions are device-independent, such as the reading and writing of character data; yet provisions have been made for device-dependent functions as well. All peripheral devices capable of dealing with character data have individual symbolic names (such as K, D, P, etc). and can be accessed using a Central I/O (CIO) routine.

A RAM data base provides access to controllers (joysticks and paddle controllers), which do not deal with character data. This RAM data base is periodically updated to show the states of these devices.

INTERRUPT PROCESSING

The interrupt system handles all hardware interrupts in a common and consistent manner. By default, all interrupts are fielded by the OS. At your discretion, individual interrupts (or groups of interrupts) can be fielded by the application program.

INITIALIZATION

The system provides two levels of initialization: power up and system reset. The OS performs power-up initialization each time the system power is switched to ON, and system reset initialization is performed each time the [SYSTEM.RESET] key is pressed.

Power-Up

The OS examines and notes the configuration of the unit whenever the system power is switched to ON. The system performs the following tasks at power up:

- Determines the highest RAM address.
- Clears all of RAM to zeros.
- Establishes all RAM interrupt vectors.
- Formats the device table.
- Initializes the cartridge(s).
- Sets up the screen for 24 x 40 text mode.
- Boots the cassette if directed.
- Checks cartridge slot(s) for diskette-boot instructions.
- Boots the diskette if directed to do so and a disk drive unit is attached.
- Transfers control to the cartridge, diskette-booted program, cassette-booted program, or blackboard program.

[SYSTEM.RESET]

Pressing the [SYSTEM.RESET] key causes the OS to perform these following tasks:

- Clears the OS portion of RAM.
- Rechecks top of RAM.
- Reestablishes all RAM interrupt vectors.
- Formats the device table.
- Initializes the cartridge(s).
- Sets up the screen for 24 x 40 text mode.
- Transfers control to the cartridge, a diskette-booted program, a cassette-booted program, or the blackboard program.

Note that [SYSTEM.RESET] does not perform all the power-up tasks listed in the power-up section.

FLOATING POINT ARITHMETIC PACKAGE

The OS ROM contains a Floating Point (FP) package that is available to nonresident programs such as ATARI BASIC.
The package is not used by the other parts of the OS itself. The floating point numbers are stored as 10 BCD digits of mantissa, plus a 1-byte exponent. The package contains these routines:

- ASCII-to-FP and FP-to-ASCII conversion.
- Integer-to-FP and FP-to-integer conversion.
- FP add, subtract, multiply and divide.
- FP log, exp, and polynomial evaluation.
- FP number clear, load, store, and move.

3 CONFIGURATIONS

The ATARI 400 and ATARI 800 Home Computers support a wide variety of configurations, each with a unique operating environment:

- Cartridge(s) may or may not be inserted
- Memory can be optionally added to the ATARI 800 Computer console in 16K increments
- Many different peripheral devices can be attached to the serial I/O bus.

The OS accounts for all of these variables without requiring a change in the resident OS itself (see Section 2). The machine configuration is checked when power is first turned on and then is not checked again, unless system reset is used. A general discussion of some of the valid configurations follows.

PROGRAM ENVIRONMENTS

The OS allows one of four program types to be in control at any point in time:

- The OS blackboard (ATARI Memo Pad) program
- A cartridge-resident program
- A diskette-booted program
- A cassette-booted program

Control choice is based upon information in the cartridge(s), upon whether or not a disk drive is attached, and upon operator keyboard inputs. The exact algorithms are discussed in detail in Section 7.

Blackboard Mode

In blackboard mode, the screen is established as a 24 x 40 text screen. Anything entered from the keyboard goes to the screen without being examined, although all of the screen editing functions are supported. Blackboard mode is the lowest priority environment. You go there only by command from a higher

priority environment, or by default, if there is no other reasonable environment for the OS to enter. For example, typing BYE in BASIC causes the OS to enter the blackboard mode. The blackboard mode can be exited by pressing the [SYSTEM.RESET] key if it was entered from a higher environment.

Cartridge

An inserted cartridge normally provides the main control after initialization is complete (for example: ATARI BASIC, SUPER BREAKOUT[TM], BASKETBALL, COMPUTER CHESS, and others. All these cartridge programs interface directly with you in some way). Although a cartridge can provide a supporting function for some other program environment, this has not yet been done. Some cartridges (particularly keyboard-oriented ones) can change environments by entering special commands (such as "BYE") to go to blackboard mode or "DOS" to enter the disk utility. Other cartridges cannot change environments. Note that a hardware interlock prevents the removal or insertion of a cartridge with the power on; this feature causes the entire system to reinitialize with every cartridge change.

Diskette Boot

The diskette may or may not be booted when the system powers up with diskette-bootable software. This paragraph assumes that a diskette boot did occur. See Section 7 for boot condition explanations.

The diskette-booted software can take control as the Disk Utility Program (DUP) does under certain conditions, or can provide a supporting function as the File Management System (FMS) does. This environment is so flexible that it is difficult to generalize on its capabilities and restrictions. The only machine requirement (other than the disk drive) is that sufficient RAM be installed to support the program being booted.

Cassette-Boot

The cassette-boot environment is similar to the diskette-boot environment, although the cassette is limited as an I/O device. It is slow and can access only one file at a time in sequence. Note that the cassette-boot facility has no relation to the use of cassettes to store high-level language programs (e.g., programs written in ATARI BASIC), nor to the use of cassettes to store data.

RAM EXPANSION

Although you can expand RAM noncontiguously in the ATARI 800 Home Computer, the OS will only recognize RAM that is contiguous starting from location 0. Installation directions are provided with the purchased RAM modules. RAM can be added until it totals 48K. After 32K, additional RAM overlays first the right-cartridge addresses (32K to 40K) and then the left-cartridge addresses (40K to 48K). Note that in cases of conflict, the inserted cartridge has higher priority and disables the conflicting RAM in 8K increments. See Section 4 for a detailed discussion of system memory.

As a result of power-up, the OS will generate two pointers that define the lowest available RAM location and the highest available RAM location. The OS and diskette or cassette-booted software will determine the location of the lowest available RAM, while the number of RAM modules and the current screen mode will determine the highest available RAM.

PERIPHERAL DEVICES

Peripheral devices of several types can be added to the system using standard cables to either the serial bus or the connectors at the front of the computer console. The most common types deal with either transmission of bytes of data (usually serial bus) or transmission of sense information (usually game controllers).

Game Controllers

The OS periodically senses (50 or 60 times per second) the standard game controllers (Paddles and Joysticks) and the values read are stored in RAM. You can plug in, remove, and rearrange these controllers at will without affecting system operation, because the system will always try to read all of these controllers.
The Driving Controllers are read, but not decoded, by the OS. Special instructions are required to read the keyboard controller (see Section 11).

Program Recorder

The ATARI 410[TM] Program Recorder is a special peripheral. It uses the serial bus to send and receive data, but does not conform to the protocol of the other peripherals that use the serial bus. The Program Recorder must also be the last device on the serial bus, because it does not have a serial bus extender connector as the other peripherals do. There can never be more than one Program Recorder connected to any system for the same reason. The system cannot sense the presence or absence of the Program Recorder, so it can be connected and disconnected at will.

Serial Bus Devices

A serial bus device conforms to the serial I/O bus protocol as defined in Section 9, but this does not include the Program Recorder. Each serial bus device has two identical connectors: a serial bus input, and a serial bus extender. Either connector can be used for either purpose. Peripherals can be "daisychained" by cabling them together in a sequential fashion. There are usually no restrictions on the cabling order because each device has a unique identifier. Where restrictions exist, they will be mentioned in Section 5.

4 SYSTEM MEMORY UTILIZATION

Memory in the system is decoded in the full 64K range of the 6502 microcomputer and there are no provisions for additional mapping to extend memory. Memory is divided into four basic regions (with some overlap possible): RAM, cartridge area, I/O region and the resident OS ROM. The regions and their address boundaries are listed below (all addresses are in hexadecimal):

```
0000-1FFF = RAM (minimum required for operation)
2000-7FFF = RAM expansion area
8000-9FFF = Cartridge B, Cartridge A (half of 16K size) or RAM
A000-BFFF = Cartridge A or RAM
C000-CFFF = Unused
D000-D7FF = Hardware I/O decodes
D800-DFFF = Floating Point Package (OS)
E000=FFFF = Resident Operating System ROM
```

Figure 4-1 6502 System Memory Map

This section will break these regions into even smaller functional divisions and provide detailed explanations of their usage.

RAM REGION

The OS and the control program share the RAM region. The RAM region can be further subdivided into the following sub regions for discussion purposes:

```
Page 0 = 6502 page zero address mode region.
Page 1 = 6502 stack region.
Pages 2-4 = OS database and user workspace.
Pages 5-6 = User program workspace.
Pages 7-XX = Bootable software area/free RAM.*
Pages XX-top of RAM = Screen display list and data.*
```

* Note that XX is a function of the screen graphics mode and the amount of RAM installed.

The paragraphs that follow describe how the OS uses RAM subregions, and presents user program recomendations.

Page 0

The architecture of the 6502 microcomputer instruction set and addressing modes gives page 0 special significance. References to addresses in that page (0000 to 00FF) are faster, require fewer instruction bytes, and provide the only mechanism for hardware indirect addressing. Page 0 should be used sparingly so that all possible users can have a portion of it. The OS permanently takes the lower half of page 0 (0000 to 007F). This portion can never be used by any outer environment unless the OS is completely disabled and all interrupts to the OS are eliminated.

The upper half of page 0 (0080 to 00FF) is available to outer environments with the following restriction: the floating point package, if used, requires 00D4 through 00FF.

Page 1

Page 1 is the 6502 hardware stack region; JSR instructions, PHA instructions, and interrupts all cause data bytes to be written to page 1. Conversely RTS, PLA, and RTI instructions all cause data bytes to be read from page 1. The 256 byte stack is adequate for normal subroutine calls plus interrupt process nesting, so no restrictions have been made on page 1 usage. It is obvious that a stack of this size is totally inadequate for deeply recursive processes or for nested processes with large local environments to be saved. So, for sophisticated applications, software maintained stacks must be implemented.

The 6502 stack pointer is initialized at power-up or system reset to point to location 01FF. The stack then pushes downward toward 0100. The stack will wrap around from 0100 to 01FF if a stack overflow condition occurs, because of the nature of the 6502's 8-bit stack pointer register.

OS Data Base

Locations 0200 through 047F are allocated by the OS for working variables, tables and data buffers. Portions of this region can be used only after you determine that nonconflict with the OS is guaranteed. For example, the printer and cassette buffers could be used if I/O operations to these devices are impossible within the controlling environment. The amount of work involved in determining nonconflict seems to be completely out of line with the benefits to be gained (except for a few trivial cases) and it is recommended that pages 2 through 4 not be used except by the OS.

User Workspace

Locations 0480 through 06FF are dedicated for outer environment use except when the floating point package is used. The floating point package uses locations 057E through 05FF.

Boot Region

Page 7 is the start of the "boot region." When software is booted from either the diskette or the cassette, it can start at the lowest free memory address (that is 0700) and proceed upward (although it can also start at any address above 0700 and below the screen display list). The top of this region defines the start of the "free memory" region. When the boot process is complete, a pointer in the data base contains the address of the next available location above the software just booted. When no software has been booted, this pointer contains the value 0700.

Screen Display List and Data

When the OS is handling the screen display, the display list that defines the screen characteristics and the current data that is contained on the screen are placed at the high address end of RAM. The bottom of this region defines the end of the free memory region and its location is a function of the screen mode currently in effect. A pointer in the data base contains the address of the last available location below the screen region.

Free Memory Region

The free memory region is all the RAM between the end of the boot region and the start of the screen region. The outer level application is responsible for managing the free memory region.

CARTRIDGES A AND B

There are two 8K regions reserved for plug-in cartridges. Cartridge B, that is the right-hand cartridge slot found only in the ATARI 800 Home Computer, has been allocated memory addresses 8000 through 9FFF. Cartridge A (the left-hand cartridge slot in the ATARI 800 Computer console, and the only slot in the ATARI 400 Computer console) has been allocated memory addresses A000 through BFFF and optionally 8000 through BFFF, for 16K cartridges. If a RAM module is plugged into the last slot such as to overlay any of these addresses, the RAM takes precedence as long as a cartridge is not inserted. However, if a cartridge is inserted, it will disable the entire conflicting RAM module in the last slot in 8K increments.

MAPPED I/O

The 6502 performs input/output operations by addressing the external support chips as memory; some chip registers are read/write while others are read-only or write-only (the ATARI Home Computer Hardware Manual gives descriptions of all of the external registers). While the entire address space from D000 to D7FF has been allocated for I/O decoding, only the following subregions are used:

```
D000-D01F = CTIA
D200-D21F = POKEY
D300-D31F = PIA
D400-D41F = ANTIC
```

Figure 4-2. Mapped I/O

RESIDENT OS AND FLOATING POINT PACKAGE ROM

The region from D800 through FFFF always contains the OS and the floating point package. Care should be taken to avoid using any entry points that are not guaranteed not to move, to allow for the possibility that another, but functionally compatible, OS can be generated in the future. The OS contains many vectored entry points at the end of the ROM and in RAM that will not move. The floating point package is not vectored, but all documented entry points will be fixed: Do not use undocumented routines found by scanning the listing. A list of the fixed ROM vectors can be found in Appendix J.

CENTRAL DATA BASE DESCRIPTION

See Appendix L.

MEMORY DYNAMICS

The free memory region is the area between the end of the boot region and the start of the screen region. As such, its limits are variable. MEMLO [02E7] defines the bottom of the free region, and MEMTOP [02E5] defines the top of the region. This section presents the conditions that cause the setup or alteration of these variables.

System Initialization Process

The OS determines the extent of the lowest block of contiguous RAM, and saves the limits. The Screen Editor is then opened, thus setting a new (and lower) value in MEMTOP. Diskette or cassette-booted software might be brought into memory, that would probably set a new (and higher) value in MEMLO (see Section 7). MEMLO and MEMTOP will define the maximum amount of free memory available when the application program finally gets control. That amount of free memory can later decrease, as described in the next paragraph.

Changing Screen Modes

The Display Handler interprets the variable APPMHI '[000E]' to contain the address below which MEMTOP cannot extend. This allows you to protect the portion of free memory space that you are using from being overwritten as a result of screen mode change. The display handler will set the screen for mode 0, update MEMTOP, and return an error status to you, if it determines that the screen memory will extend below APPMHI as a result of a screen mode change. In other cases the Display Handler effects the desired mode change and updates MEMTOP.